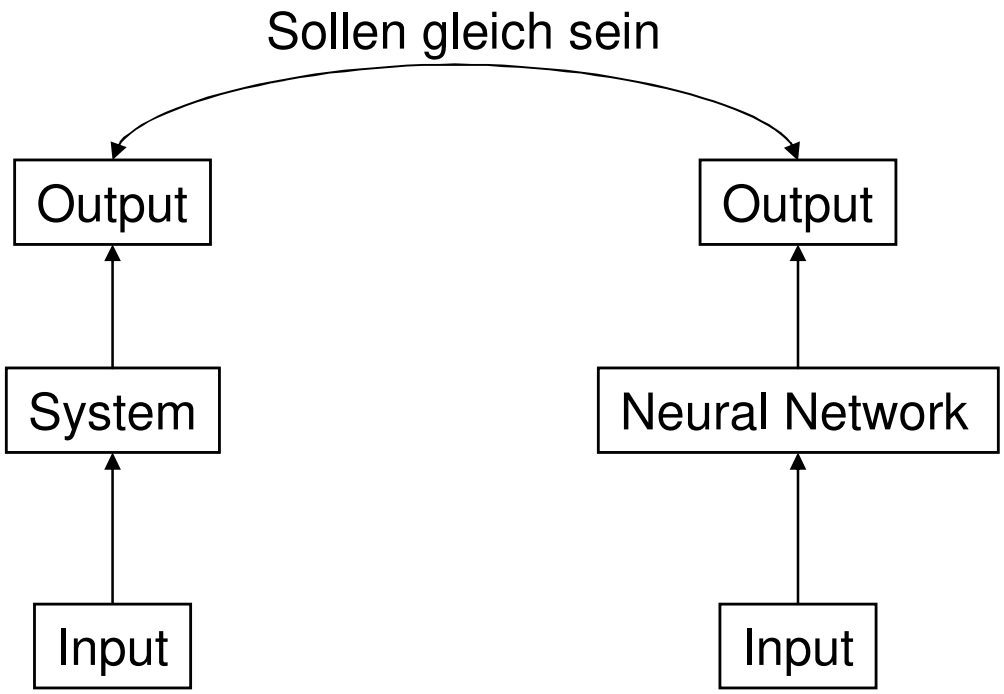
Supervised Learning

Funktionsapproximation



Messen des Fehlers

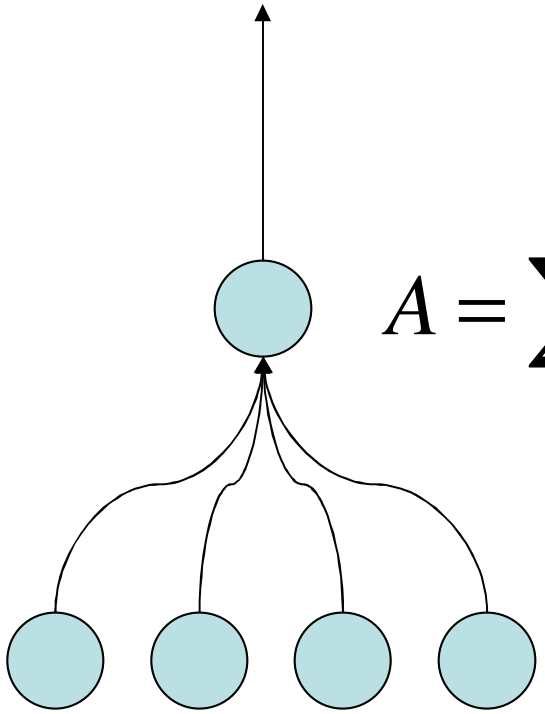
$$\Psi_{error} = \left\langle A_{ist} - A_{soll} \right\rangle_{alle\ daten}$$

Ein Lineares System

Output A

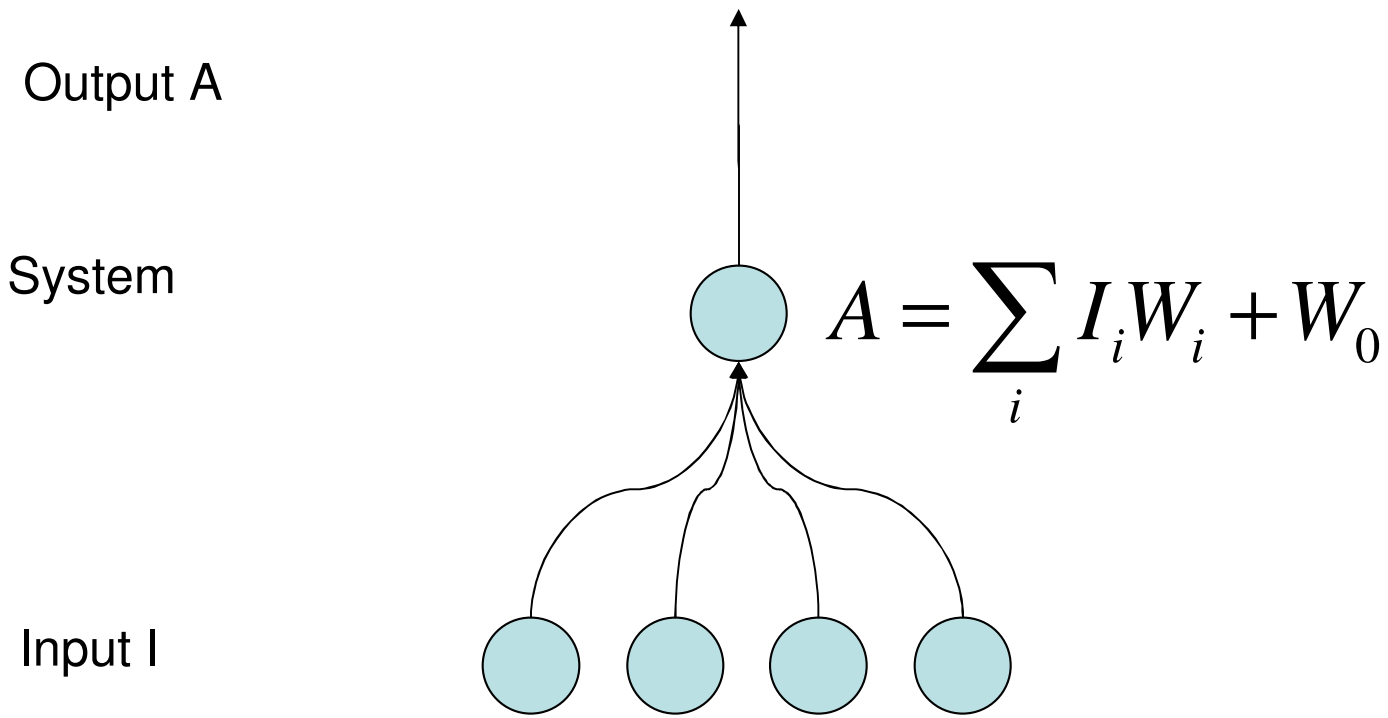
System

Input I



$$A = \sum_i I_i W_i + W_0$$

Die Delta-Regel

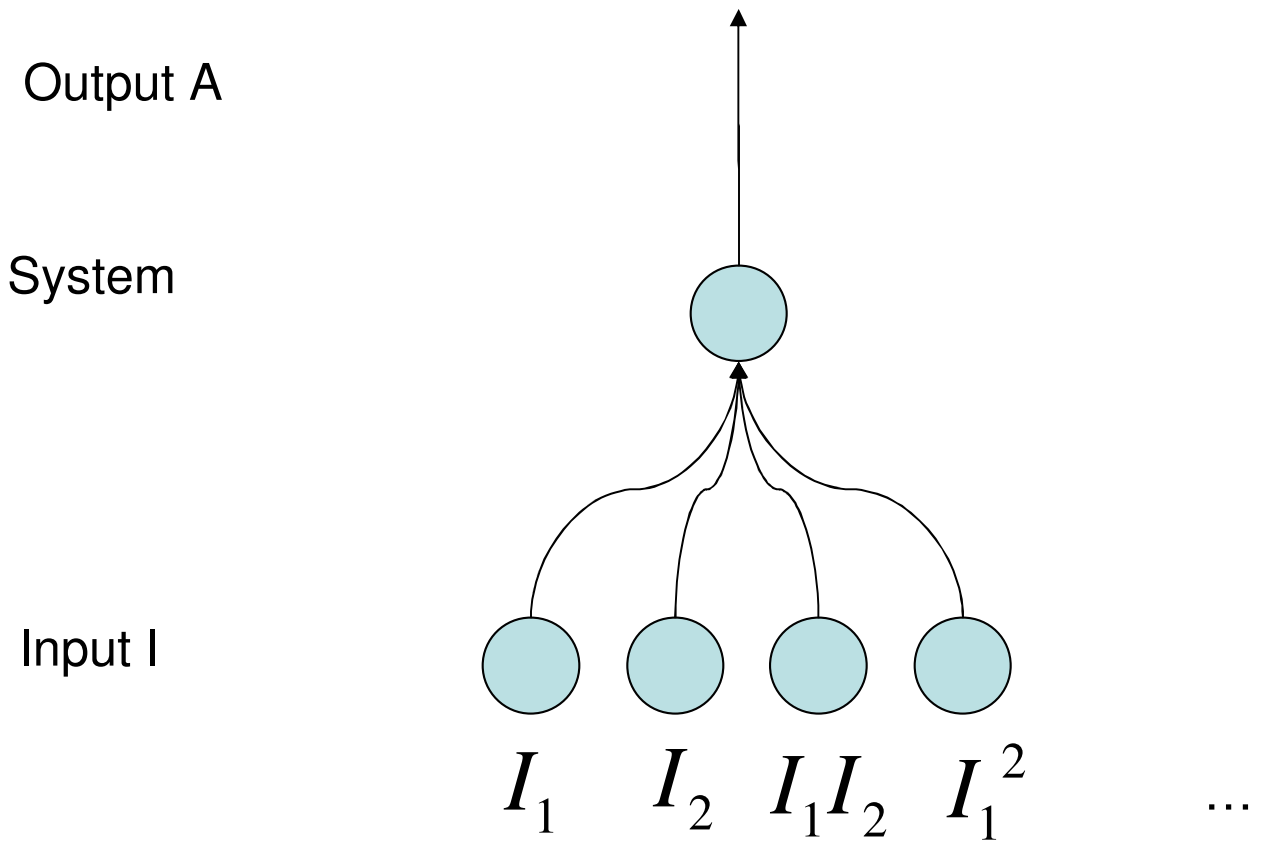


$$\Delta W_i = \varepsilon \left(A - A_{soll} \right) I_i$$

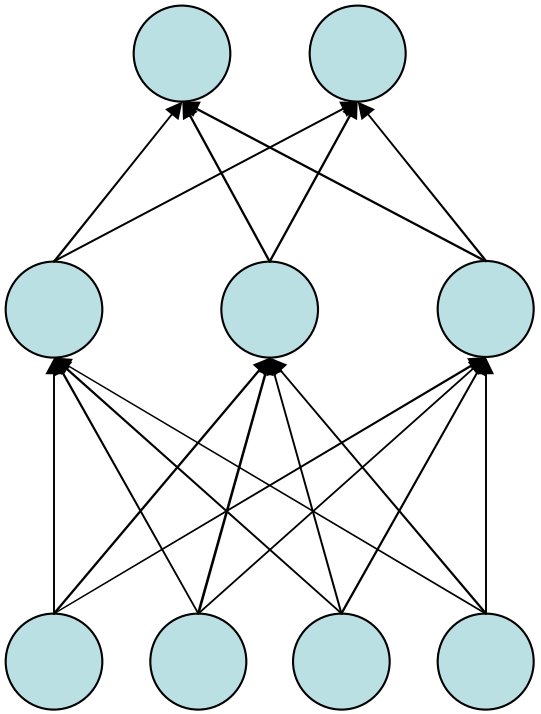
Xor- ein unlösbares Problem

input		output
0	0	1
1	0	0
0	1	0
1	1	1

Support Vector Machines



Multilayer-Perceptrons

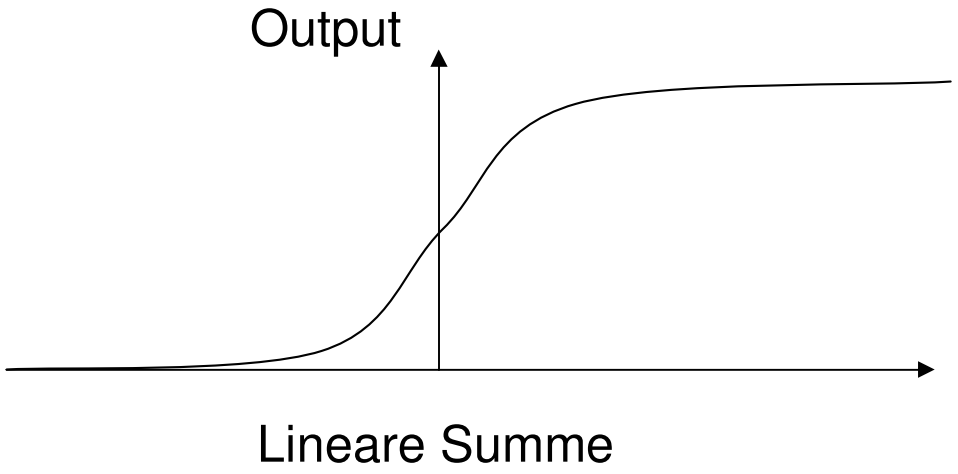


$$A_j = f\left(\sum_i H_i W_i + W_0\right)$$

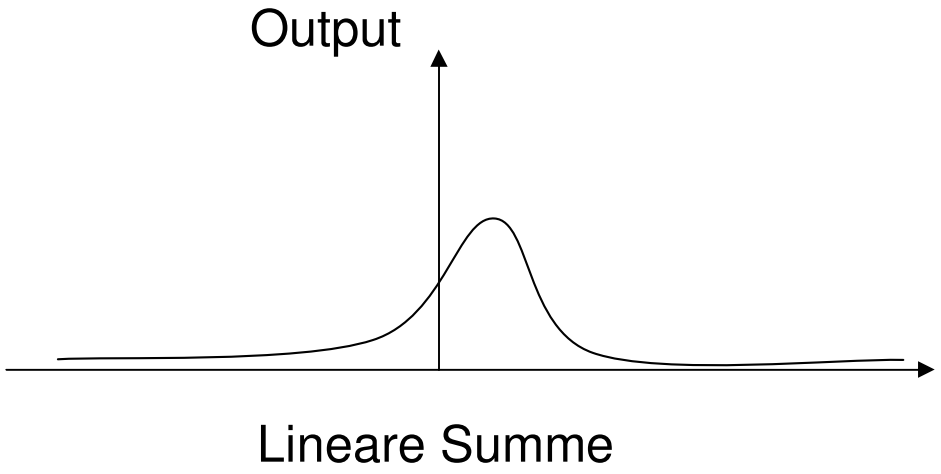
$$H_j = f\left(\sum_i I_i W_i + W_0\right)$$

I

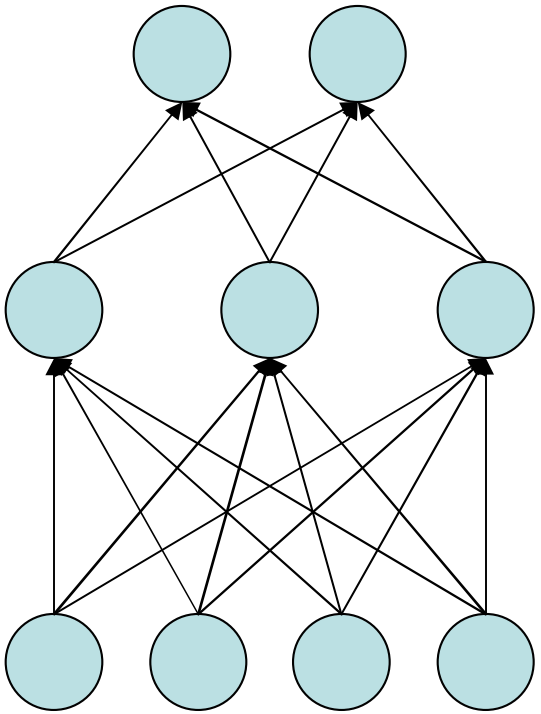
Nichtlinear funktion – multilayer perceptron



Nichtlinear funktion – radial basis function networks



Problem beim Lernen



$$A_j = f\left(\sum_i H_i W_i + W_0\right)$$

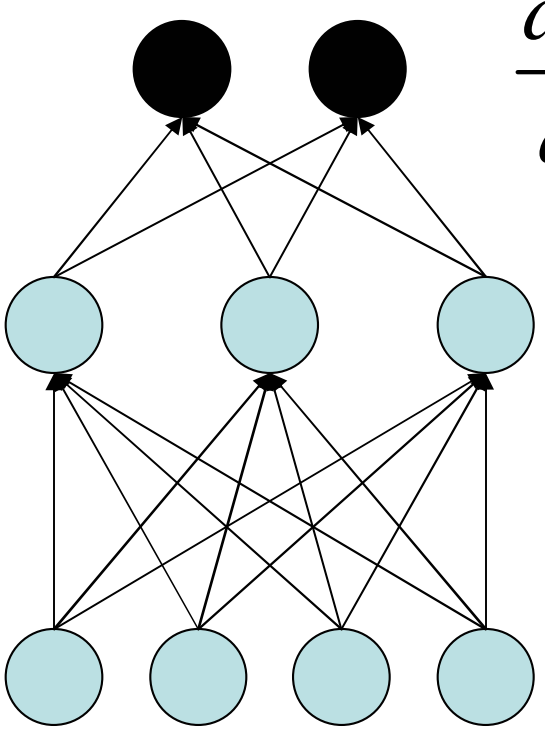
$$H_j = f\left(\sum_i I_i W_i + W_0\right)$$

I

Wie sollen wir die W ändern, so dass der output möglichst gut ist?

Methode

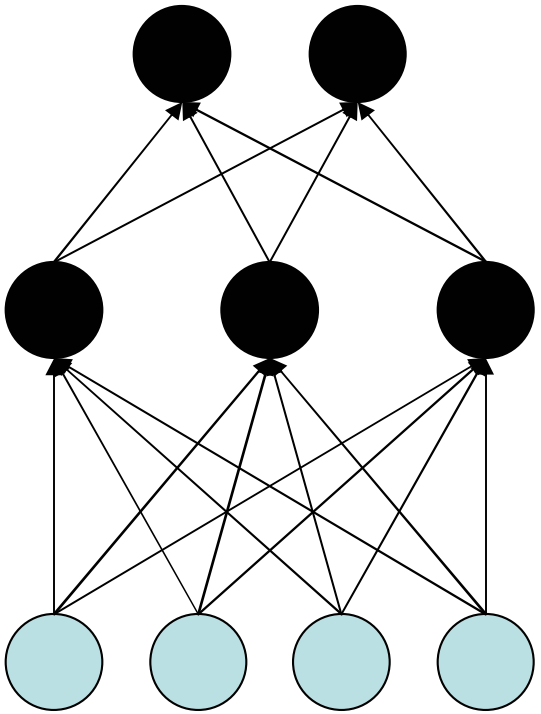
Werbos, Rummelhart und McClelland



$$\frac{d\Psi}{dA}$$

Backpropagation of error

Werbos, Rummelhart und McClelland



$$\frac{d\Psi}{dH_i} = \sum_{\text{postsynaptic neurons } j} \frac{d\Psi}{dA_j}\frac{dA_j}{dH_i}$$

$$= \sum_{\text{postsynaptic neurons } j} \frac{d\Psi}{dA_j} f'\left(A_j\right) W_{ij}$$

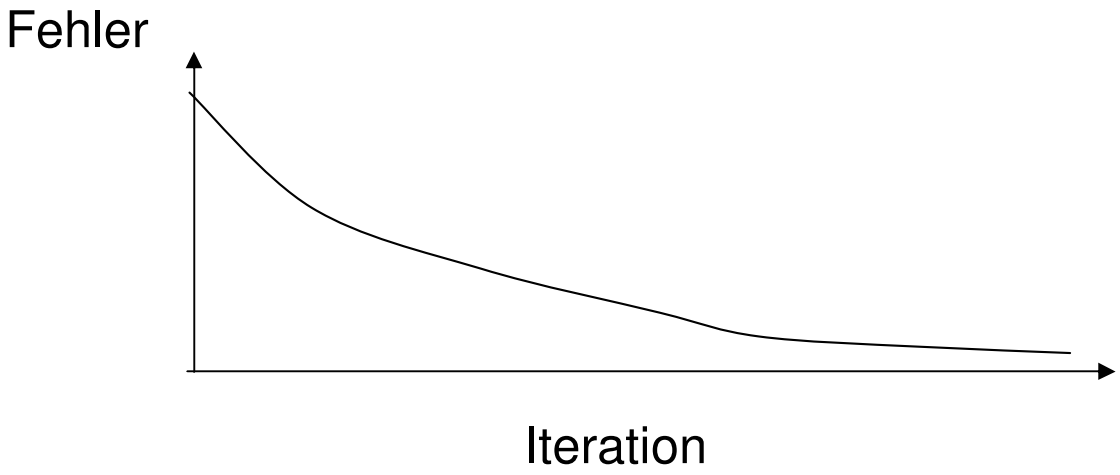
Warum so erfolgreich

- Ableiten ist schnell O(Zahl der Verbindungen)
- Kann (theoretisch) alle kontinuierlichen Funktionen approximieren

Iteratives Lernen

$$\Delta W_i = -\varepsilon \frac{d\Psi}{dW_i}$$

Gradientenabstieg

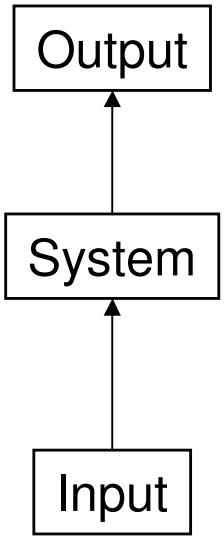


Die richtigen Ws schneller finden

Moment

$$\Delta W_i = -\varepsilon \frac{d\Psi}{dW_i} + \beta \Delta W_{i\,last\ iteration}$$

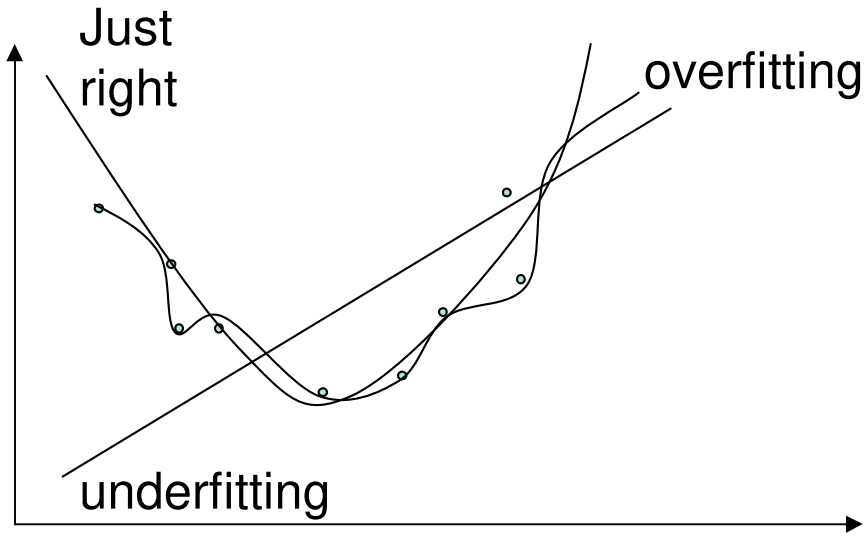
Typische Anwendungen



Typ	Beispiel Input	Output
Klassifizierung	Radarsignale	Ist es eine Miene?
	Geschriebene Sprache	Lautschrift
	Gesprochene Sprache	Geschriebene Sprache
Approximierung	Video	Speed
Vorhersage	Aktienmarkt gestern	Aktienmarkt heute

Generalisierung

Trainingsdaten	Neue Testdaten
Sehr gut	Echt schlecht



Besseres Generalisieren

Messe Kompliziertheit der gefitteten Funktion

$$\Psi_{total} = \Psi_{error} + \Psi_{regularisierung}$$

Möglichkeiten:
- Höre nach wenigen iterationen auf
- Benutze möglichst wenig Neurone
- Benutze kleine Gewichte
- Suche Gewichte, so das sich der Fehler möglichst wenig ändert wenn die Gewichte sich ändern (e.g. Schmidhuber)

Schlechtes Skalieren

Multilayer Perceptrons unmöglich für mehr als etwa 100 inputs

Für jeden Stimulus bekommt man nur den Fehler gesagt

Mit diesem muss man dann die Gewichte bestimmen

Wenn das Problem schwieriger wird braucht man viel mehr gewichte

Kein gutes Modell für Lernen von Tieren

- Selten Supervised
- Objekterkennung ohne Namen
- Alternative: Unüberwachtes Lernen – Lernen durch beobachten

Zusammenfassung

- Backpropagation = schnelle Ableitung
- Erfolgreich für technische Applikationen
- Generalisation wird hinzugefügt
- Probleme für grosse Systeme

Useful tutorial

goto

http://diwww.epfl.ch/mantra/tutorial/english/

I recommend the character recognition tutorial